REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE 11 June 1996	3. REPORT TYPE AND DATES COVERED PROFESSIONAL PAPER

4. TITLE AND SUBTITLE

PROVIDING COMMON MUNITION MODELS VIA AN ORDNANCE SERVER

5. FUNDING NUMBERS

6. AUTHOR(S)

John DiCola, David Mutschler, Lawrence Ullom, Alexandra Wachter

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(ES)

COMMANDER
NAVAL AIR WARFARE CENTER AIRCRAFT DIVISION
22541 MILLSTONE ROAD
PATUXENT RIVER, MARYLAND 20670-5304

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

COMMANDER
NAVAL AIR SYSTEMS COMMAND
1421 JEFFERSON DAVIS HIGHWAY
ARLINGTON, VA 22243

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

In Distributed Interactive Simulation (DIS) exercises, it is often required that simulated entities interact on an equal basis. When different simulators use different models for the same munitions, combat between the simulated entities may be skewed in favor of one simulation over another. Thereby, the validity of exercise data might be lessened and the worth of the exercise reduced.

Using common munitions models eliminates this problem. One approach toward providing these models is to use an establish procedure or object class library where an entity's simulator would also simulate the ordnance it fires. However, since this library must be compiled and bound to the platform simulation, this approach may lead to integration and performance issues.

This paper describes an alternative: an ordnance server. the ordnance server acts as a common repository that interfaces directly with the network. Once a munitions is fired, the server assumes control and simulates it apart from its launching platform. Given available information, the server employs the appropriate weapon model. The server simulates weapon flyout, status,trajectory, impact and other munitions attributes.

Since the server may be located apart from simulation platforms, processing load may be better distributed. Also, by operating within the DIS protocol, the ordnance server provides no additional network load. Furthermore, it can be distributed to different sites about the network. Lastly, by using ordnance servers near target sites, latency between a weapon detonation and its effect on a target can be substantially lessened.

14. SUBJECT TERMS

Distributed Interactive Simulation (DIS); ordnance

15. NUMBER OF PAGES

5

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	N/A

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102

19960620 109

DTIC QUALITY INSPECTED 1

Encl (1)

DEPARTMENT OF THE ARMY
U.S. ARMY RESEARCH INSTITUTE
5001 EISENHOWER AVENUE
ALEXANDRIA, VA 22333-5600

REPLY TO ATTENTION OF

11 JUN 1996

PERI-II

MEMORANDUM FOR Team Leader, Technical Publishing Team, Naval Air Warfare Center Aircraft Division, 22541 Millstone Road Patuxent River, MD 20670-5304

SUBJECT: Request for Release of Technical Information

1. Reference request for review dated 10 June 96, of the paper titled "Providing Common Munitions Models Via an Ordnance Server" for presentation at the 18th Interservice/Industry Training Systems and Education Conference on 3-6 December 1996.

2. The undersigned reviewed the referenced paper and, as manager of a directly related program of work, approves it for public release and presentation at the conference.

Franklin L Moses
FRANKLIN L. MOSES
MDT2 Program Manager

PROVIDING COMMON MUNITION MODELS VIA AN ORDNANCE SERVER

John DiCola, David Mutschler, Lawrence Ullom, Alexandra Wachter
ACETEF/MFS, Naval Air Warfare Center -- Aircraft Division
Patuxent River, MD

ABSTRACT

In Distributed Interactive Simulation (DIS) exercises, it is often required that simulated entities interact on an equal basis. When different simulators use different models for the same munitions, combat between the simulated entities may be skewed in favor of one simulation over another. Thereby, the validity of exercise data might be lessened and the worth of the exercise reduced.

Using common munition models eliminates this problem. One approach toward providing these models is to use an established procedure or object class library where an entity's simulator would also simulate the ordnance it fires. However, since this library must be compiled and bound to the platform simulation, this approach may lead to integration and performance issues.

This paper describes an alternative: an ordnance server. The ordnance server acts as a common repository that interfaces directly with the network. Once a munition is fired, the server assumes control and simulates it apart from its launching platform. Given available information, the server employs the appropriate weapon model. The server simulates weapon flyout, status, trajectory, impact, and other munition attributes.

Since the server may be located apart from simulation platforms, processing load may be better distributed. Also, by operating within the DIS protocol, the ordnance server provides no additional network load. Furthermore, it can be distributed to different sites about the network. Lastly, by using ordnance servers near target sites, latency between a weapon detonation and its effect on a target can be substantially lessened.

BIOGRAPHIES

John DiCola
Mr. DiCola is a Simulation Engineer employed at the ACETEF/Manned Flight Simulator in the Naval Air Warfare Center at Patuxent River, MD. He has a BSEE from Virginia Tech., and an MS in Computer Science from the Florida Institute of Technology. His interests are distributed simulation and artificial intelligence.

David Mutschler
Mr. Mutschler is a Computer Engineer employed at the ACETEF/Manned Flight Simulator in the Naval Air Warfare Center Aircraft Division at Patuxent River, MD. He has received a BA from Rutgers University, a Masters of Engineering from Penn. State University, and an MS from Temple University where he is currently a Ph.D. candidate in Computer and Information Science. His interests include distributed computing, simulation, software engineering, and computer graphics. He is a member of the ACM, IEEE, and IEEE/CS.

Lawrence Ullom
Mr. Ullom is an Electronics Engineer employed at the Air Combat Environment Test and Evaluation Facilities (ACETEF) / Manned Flight Simulator in the Naval Air Warfare Center at Patuxent River, MD. He has a BSEE from West Virginia Institute of Technology. His interests include networking, distributed systems, and simulation. He has served in several DIS subworking groups including Interface and Computing Architectures.

Alexandra Wachter
Mrs. Wachter is an Electronics Engineer employed at the ACETEF/Manned Flight Simulator in the Naval Air Warfare Center at Patuxent River, MD. She has a BEE from the Georgia Institute of Technology. She is the Distributed Simulation Team Leader in the Manned Flight Simulator laboratory.

PROVIDING COMMON MUNITION MODELS VIA AN ORDNANCE SERVER

John DiCola, David Mutschler, Lawrence Ullom, Alexandra Wachter
ACETEF/MFS, Naval Air Warfare Center -- Aircraft Division Patuxent River, MD

PROBLEM STATEMENT

Need For Common Munitions Models

In the Distributed Interactive Simulation (DIS) community, the lack of validated, DIS compatible, high fidelity munition models has been a major concern. Models of varying fidelity have been used together in simulation exercises, resulting in questions of exercise validity based on inconsistent munition behavior and effectiveness across platform simulations. In other words, persistent fidelity variations between munition simulations have caused many participants to claim that the exercise did not produce a fair fight.

One problem is that many munition simulations are integrated directly into the launching platform's simulation. Different models developed for different simulations are developed independently and are constructed using different design criteria and fidelity requirements. Hence, it becomes difficult to determine the "best" model. Moreover, because of different interfaces, data requirements, and levels of coupling with their launching simulations, it may also be difficult to extract a munition model from one simulation and integrate it into another.

To solve this (and other) problems, the Manned Flight Simulator (MFS) laboratory of the Air Combat Environment Test and Evaluation Facility (ACETEF) designed and implemented the Ordnance Server (OS). The OS is separate and distinct from any launching simulation. It can be accessed remotely by any simulation within an exercise. Thereby, each simulation can access a uniform set of munition models.

Use Of Libraries Is Inadequate

Another method of sharing models is using a procedure or object library of common munition routines. Simulations would link with the appropriate routines for the munitions they require.

Unfortunately, while a library would permit different simulations to share munition models, it introduces several other problems. It may be difficult to link the needed routines into the simulations because of the variety of munition model interfaces. This is especially true for legacy systems. In addition, configuration management of the routines can be difficult since software must be made available to different sites. Furthermore, each time a munition routine is updated, all of the simulations that use that routine must be relinked to take advantage of any improvements. Lastly, since the munition model could execute on the same processor as the simulation, the munition model would contend for that processor's resources.

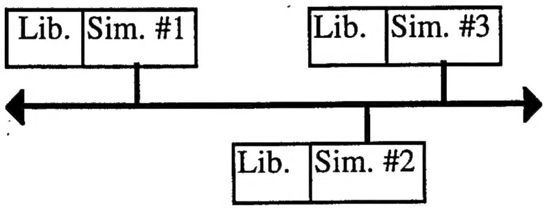

Figure 1. Simulation w. Common Libraries

This paper will show that an ordnance server provides common munition models without the drawbacks of a procedure or object class library. Furthermore, since the ordnance server uses standard DIS messages required for the firing of any ordnance, there is no additional loading of the network.

SOLUTION STATEMENT -- ORDNANCE SERVER AS SUPERIOR SOLUTION

The ordnance server concept offers many advantages over a library of munition routines. These advantages include: ease of integration, testability, ease of configuration management, distribution of processing load, and reduction of network latency.

Integration to the ordnance server is provided by the standard DIS Fire PDU. When the server receives this message, it utilizes the given information to discern the appropriate munition model and execute it appropriately. This information is provided regardless of the munition's launching platform type. Thereby, commonality is ensured and integration into existing DIS simulations is straightforward.

Greater testability is available because the ordnance is simulated using a separate executable. The munitions can be tested, verified, and validated, independently of a launching platform. Furthermore, there should be no requirement to revalidate either a platform or munition model when any change is made to the other. This separation of the ordnance models from the launching vehicle models also facilitates the addition of new models and can reduce the overall testing effort required.

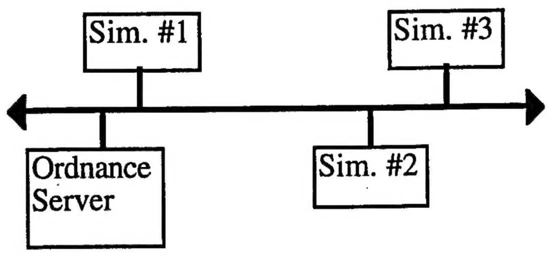

Figure 2. Simulation with an Ordnance Server

The use of a server also facilitates configuration management since all munition models are interfaced via the server alone. Proliferation of software to each of the simulation sites is not necessary.

An ordnance server's ability to run on any supported machine on the network during a DIS simulation provides many benefits. The munition server can run on the same hardware as the launching platform, or can run independently on a separate machine. Thus, it does not impact the processing power available to any of the entity simulations.

An ordnance server is adaptable. It can be configured to operate for selected munitions, types of launching entities, or individual entities. Thereby, processing load between ordnance simulations can be further distributed.

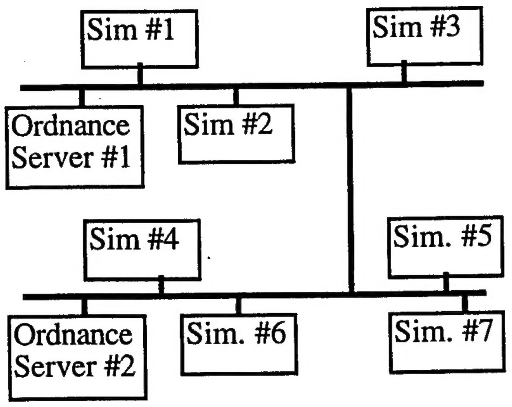

Figure 3. A Distributed Ordnance Server

In addition, distributing the ordnance server can be used to reduce network loading in many cases. One approach is to collocate an ordnance serve and the target entity (either on the same LAN or even the same computer). By significantly reducing the transmission length between the munition simulation and it's target, network latency can be drastically reduced. For fast, highly dynamic targets, this could alter the end-game outcome since the targeted entity would have more timely updates of munition location (for avoidance) or impact (for damage assessment). It has been shown in other research (2) that network latency is a dominant variable in the interactivity limits of distributed simulation.

Lastly, the ordnance server only utilizes standard DIS PDUs. No extra messages are generated. Hence, utilization of an ordnance server adds no additional loading onto the network. Furthermore, increasing the number of munition servers will not, by itself, increase the amount of traffic on a network, and the increased bandwidth required by additional munition models will be no different than if standard munition simulations were used. Of course, each munition server should be configured to avoid overlap to prevent multiple munition simulations from being erroneously generated given a single Fire PDU.

IMPLEMENTATION

The Ordnance Server developed at ACETEF/MFS has been successfully used in many exercises and demonstrations. Several types of Tactical Air Combat Training Systems (TACTS) munition models have been incorporated into the OS to increase the fidelity of the munition simulations. The OS was used to supply these validated missile models for several events including tactical air exercises requiring realistic air-to-air munitions. The OS was also demonstrated at I/ITSEC 95 where it was used in conjunction with the Simulated Warfare Environment Generator (SWEG) to simulate OPFOR surface to air munitions.

ORDNANCE SERVER DESCRIPTION

Message Flow

DIS uses three types of PDUs to describe tracked munition simulation: Fire, Entity State, and Detonation PDUs. Should the munition emit any signals (as for radar), Emission PDUs may also be required. The Fire PDU provides munition launch information, including munition type, launching entity ID, munition entity ID, and a target entity ID (if applicable). Once the munition is launched, Entity State PDUs are issued to describe the munition's location and orientation. At the termination of the munition's flight, a Detonation PDU is issued, regardless of the reason for termination. The Detonation PDU describes the reason for termination, the termination location, and the target entity ID if applicable. The target simulation is then responsible for determining the effects, if any, of the missile detonation.

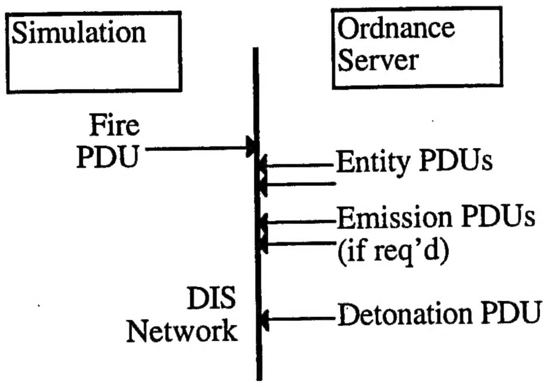

Figure 4. DIS Message Flow

When not using an ordnance server, all three PDUs are normally output by the same simulation. When the OS is used, the launching entity only outputs a Fire PDU. The OS then assumes control of the munition. It uses Entity State PDUs to report it's flight path and Emission PDUs to report it's emissions (if any). It is also the OS's responsibility to produce the detonation PDU when the munition's flight ends.

Configuring the Ordnance Server

The OS is extremely flexible and can be adapted to work in a wide variety of situations. The keys to this flexibility are the configuration file and the graphical user interface (GUI). The configuration file permits speedy initialization of the OS and is loaded at startup. The GUI permits change of OS parameters at run-time. These modifications may also be saved into a configuration file.

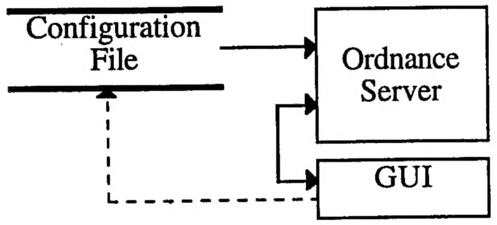

Figure 5. OS Configuration

Amidst the configuration data, the most important information is the site, application, and entity ID that distinguish the platforms to be served by the OS. Through the use of wildcards, the OS can be configured to serve a single entity, all entities from a specific site and application, or all entities from a specific site. This allows the user to set up multiple OS's in the optimal configuration that best matches resources and requirements.

The configuration file also contains the name of the terrain database file that is used to check for ground impacts. The OS currently uses ModSAF Compact Terrain Database (CTDB) type databases.

The data also contains the entity type to fly-out model mappings. For example, the OS could be set up to use a single AIM-9 flyout model whenever any variant of the AIM-9 is launched.

Fuse and warhead types for each munition may also be specified. Lastly, the emissions produced by a munition's radar (if any) can also be specified.

Munition Models

The OS currently includes two types of munition models: the generic models and the detailed external models. With the generic munition models, the user can control several parameters, including drag, weight, burn rate, and guidance method. Each of these parameters can be modified via the GUI, and saved to the configuration file.

The other type of munition model supported is the detailed external model. These models are developed externally to the ordnance server and integrated into the OS via the model interface adapter (MIA). The model interface adapter must be specifically tailored for each external model. The MIA serves as a bi-directional interface that inputs and outputs munition and synthetic environment data from the model in the same manner and format as the model's original simulation executive. Thereby, integration of the model into the OS is greatly facilitated.

There are currently two types of external models incorporated into the OS: Tactical Air Combat Training System (TACTS) models, and the Tomahawk Trajectory Generation Tool. The TACTS models are validated, real-time, high fidelity missile models used for training Naval air personnel on the Tactical Combat Training Ranges. They feature a well-defined interface that is standard across weapon types. This greatly simplified the addition of TACTS models to the OS because a single model interface adapter could work for several munition models.

TACTS Models	**Common Name**
AA-2B, 2D	Atoll
AA-8A, 8C	Aphid
AA-10A, 10B, 10C	Alamo
AA-11	Archer
AIM-7F, 7M, 7E2	Sparrow
AIM-9G, 9H, 9L, 9M, 9P3	Sidewinder
AIM-54A, 54C	Phoenix
AIM-120	AMRAAM
FIM-92	Stinger
R-550	Magic

SA-2	Guideline
SA-3	Goa
SA-4	Ganef
SA-5	Gammon
SA-6	Gainful
SA-7	Grail
SA-8	Gecko
SA-9	Gaskin
SA-16	
SA-N-3	Goblet

Table 1. TACTS Models in the OS

The Tomahawk Trajectory Generation Tool was developed by the Dahlgren Division of the Naval Surface Warfare Center, and is used to perform Tomahawk munition fly-outs. Additional TACTS models are currently being added, together with other types of external models.

CONCLUSION

An ordnance server provides many benefits over and above those benefits provided by a library of common munition models. While both methods provide for common munition modeling, the ordnance server further supports testability, ease of integration, configuration management, and distributed process loading. Distribution of the ordnance server about the network can also minimize network latency.

Integration of an ordnance server is facilitated by the use of the Fire PDU as its interface. Since the OS requires no additional messages, there is no addition to network loading. Furthermore, the OS is highly configurable to meet a variety of exercise requirements. These include munition simulation for one or all entities in an exercise, model mappings, and warhead selection.

In past efforts, the ordnance server implemented by ACETEF/MFS has proven to be a highly successful tool. Given its versatility, it is being adapted for use with the High Level Architecture (HLA) whereby it will be used for STOW '97.

REFERENCES

1. DiCola, John; Fischer, Peter; Mutschler, David; Ullom, Lawrence. "Improving Munition Simulation Fidelity Through Use of an Ordnance Server", Proceedings of the AIAA Flight Simulation Technologies Conference. July, 1996 (to be published)

2. Foster, Lester and Feldmann, Peggy. "The Limitations of Interactive Behavior for Valid Distributed Interactive Simulation", 13th DIS Workshop on Standards for the Interoperability of Distributed Simulations (Report 95-13-027). Institute for Simulation and Training. Orlando, FL. pg. 175-189

3. Ullom, Lawrence and Fischer, Peter. "Using an Ordnance Server to Provide Validated Weapon Models to MODSAF", Proceedings of the 6th Computer Generated Forces and Behavioral Representation Conference. 1996 (to be published)